ICT System Description for the 2006 TC-STAR Run #2 SLT Evaluation

Zhongjun He, Yang Liu, Deyi Xiong, Hongxu Hou, Qun Liu

Institute of Computing Technology
Chinese Academy of Sciences
No.6 Kexueyuan South Road, Haidian District
P.O. Box 2704, Beijing, China, 100080
{zjhe, yliu, dyxiong, hxhou, liuqun}@ict.ac.cn

Abstract

This paper describes systems participated in 2006 TC-STAR Run #2 SLT Evaluation of Institute of Computing Technology, Chinese Academy of Sciences. We developed three systems based on different techniques: system Confucius based on phrase, system Lynx based on tree-to-string alignment template and system Bruin based on BTG (Bracketing Transduction Grammar). These three systems share the same phrase-based translation model and language model. We also focused on improving phrase extraction and training large data. We participated in both the ASR and Verbatim tasks of Chinese Mandarin to English translation. The results and the conclusion are given.

1. Introduction

This paper describes systems participated in 2006 TC-STAR Run #2 SLT Evaluation of Institute of Computing Technology, Chinese Academy of Sciences. The primary system Confucius is a phrase-based SMT (statistical machine translation) system. Additionally, we developed other two systems, one system named Lynx is based on tree-to-string alignment template, and the other one named Bruin is based on BTG. In the following sections, we will firstly describe data preparing, and then the translation models of three systems. Experiments will be reported in section 4.

2. Data Preparing

This section describes how we prepare the training data for our systems, including data preprocessing, word alignment and phrase extraction and scoring.

2.1. Data Preprocessing

Data preprocessing is a very important step in machine translation system, which can impact the word alignment and translation quality. In our experiments, following steps are performed to the training data:

- Tokenization: This will transform Chinese characters into Chinese words, and separate punctuation from words in both Chinese and English sentence
- True case mapping: We check the beginning words of English sentences in the training corpus, if its lowercase version occurs more often, then we map the uppercase to its lowercase
- SBC case to DBC case: Numbers and English words often occurs in SBC case in Chinese, such as "１２３", "ＡＢＣ", which are replaced by its DBC case "123", "ABC" in this step

2.2. Word Alignment

We first run GIZA++ (Och and Ney, 2000) to IBM model 4 in both translation directions to get a initial word alignment, and then apply "grow-diag-final" method (Koehn et al., 2003) to refine it.

Firstly, we intersect the two alignments obtained by running GIZA++, e.g., Chinese to English and English to Chinese, and get a high-precision alignment. Then the intersection alignment is "growing" iteratively by adding potential alignments, which exist in the union of the two alignments. We check the neighbors of the intersection points in alignment matrix, including left, right, up, bottom and the diagonally directions, if either of the words linked by the potential alignment is not aligned previously, the potential alignment is added. This operator is done until no more neighbors can be added. In the final step, potential alignments connect unaligned words are added.

2.3. Phrase Extraction

Bilingual phrases can be learned from word aligned parallel corpus. As is common in most phrase-based SMT systems, we consider bilingual phrase as a pair of source and target words sequences, with the following constrains:

1. the words should consecutive in both source and target sentences
2. the word level alignment of bilingual phrase should consists with the alignment matrix

The consistency means that the words of the bilingual phrase can only be aligned to each other, and not to any other words outside.

Our extraction method is very similar to Och (2002). For a word aligned sentence pair, we enumerate all the consecutive words sequences of English sentence, and for each English phrase, find the corresponding Chinese words according to alignment matrix, if it satisfies the two constrains above, a bilingual phrase is extracted. In addition, in order to extract more phrases, such a bilingual phrase can be

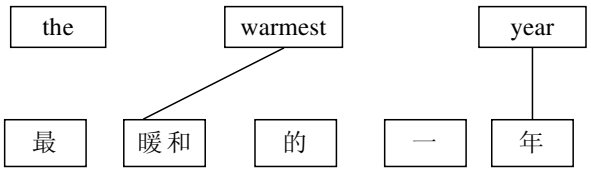


Figure 1: Word aligned sentence pair

English	Chinese
warmest	暖和
warmest	暖和 的
warmest	暖和 的 一
warmest	最 暖和
warmest	最 暖和 的
warmest	最 暖和 的 一
the warmest	暖和
the warmest	暖和 的
the warmest	暖和 的 一
the warmest	最 暖和
the warmest	最 暖和 的
the warmest	最 暖和 的 一
warmest year	暖和 的 一 年
warmest year	最 暖和 的 一 年
year	年
year	一 年
year	的 一 年
the warmest year	最 暖和 的 一 年
the warmest year	暖和 的 一 年

Table 1: Bilingual phrases extracted from the example

extended at Chinese side since "NULL" alignment is allowed, which means a word aligned to nothing. For the same English phrase, we extend the corresponding Chinese phrase to both left and right, if the added Chinese word is not aligned, and the new phrase satisfies our definition, it is extracted as a bilingual phrase. This is done iteratively until the extended word is aligned.

See Figure 1 for an illustration. By extending the unaligned word, a lot of phrases are extracted from the example as show in Table 1. Theoretically, we can extract phrases of arbitrary length, but experiments show that longer phrases don't yield better translation quality (Koehn et al., 2003). Therefore, the length of phrases is limited from 1 word to 7 words in our experiment. Please notice that the word is considered as a special phrase in our system.

2.4. Phrase Probability

Using relative frequency, we define phrase translation probability as:

$$p(\widetilde{f}|\widetilde{e}) = \frac{N(\widetilde{f},\widetilde{e})}{\sum_{\widetilde{f}'} N(\widetilde{f}',\widetilde{e})} \quad (1)$$

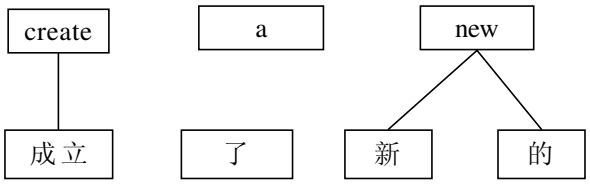
where $N(\widetilde{f},\widetilde{e})$ denotes the total number of bilingual phrase $(\widetilde{f},\widetilde{e})$ occurred in the training corpus. If one occurrence of $\widetilde{e}$ aligned to N possible foreign phrases, each of them contributes to $N(\widetilde{f},\widetilde{e})$ with $1/N$.

$p(\widetilde{f}|\widetilde{e})$ denotes the probability of translating phrase $\widetilde{e}$ to phrase $\widetilde{f}$ as a whole. additionally, we want to know how well the words of phrase $\widetilde{e}$ translate to the words of phrase $\widetilde{f}$. Following the description in (Koehn et al., 2003), given a bilingual phrase (f_1^J, e_1^I) and its alignment a, the lexical weight is defined as:

$$lex(f_1^J|e_1^I,a) = \prod_{j=1}^{J} \frac{1}{|\{i|(j,i)\in a\}|} \sum_{\forall(j,i)\in a} p(f_j|e_i) \quad (2)$$

See Figure 2 for an illustration.

For computing phrase lexical weight, we should know the word level alignment of bilingual phrases, as well as the word translation probability. When extracting phrases from the training corpus, the alignment information is reserved, moreover, a special token "NULL" is added to each English sentence and aligned to unaligned foreign words, and then the word translation probability can be computed according to equation (1).



$$lex(\text{成立 了 新 的}|create\ a\ new)$$
$$= p(\text{成立}|create) \times p(\text{了}|NULL)$$
$$\times \frac{1}{2}(p(\text{新}|new) + p(\text{的}|new))$$

Figure 2: Example for computing lexical weight

3. System Overview

We have developed three systems based on different techniques; they are Confucius (based on phrase), Lynx (based on tree-to-string alignment template), Bruin (based on BTG). In this section, we will describe them in detail.

3.1. Confucius

Confucius is a phrase-based decoder. Phrase-based machine translation does well with local context dependency and short idioms, which is the start-of-the-art method in MT society. Following other researchers work, we developed a phrase-based system.

3.1.1. Translation Model

As described in (Och and Ney, 2002), we employ a log-linear approach, which is a direct translation model:

$$Pr(e_1^I|f_1^J) = p_{\lambda_1^M}(e_1^I|f_1^J)$$
$$= \frac{\exp[\sum_{m=1}^{M} \lambda_m h_m(e_1^I, f_1^J)]}{\sum_{e'^I_1} \exp[\sum_{m=1}^{M} \lambda_m h_m(e'^I_1, f_1^J)]} \quad (3)$$

Among all possible translations, we select the one with highest probability:

$$\hat{e}_1^I = \underset{e_1^I}{\operatorname{argmax}} Pr(e_1^I|f_1^J) \quad (4)$$

Six features are used in our translation model:

- Phrase translation probability $p(\widetilde{e}|\widetilde{f})$
- Inverse phrase translation probability $p(\widetilde{f}|\widetilde{e})$
- Phrase lexical weight $lex(\widetilde{e}|\widetilde{f})$
- Inverse phrase lexical weight $lex(\widetilde{f}|\widetilde{e})$
- English language model $lm(e_1^I)$
- English sentence length penalty I

3.1.2. Deal With Named Entity

Instead of building a special model for named entity, we adopt a simple way to deal with them. Number and time phrases are translated according to rules. It is easy to translate "一百三十五" to "135", "三月 六日" to "March, the 6th". Those phrases are put into the phrase table with a certain probability (experience value), together with other phrases extracted from training corpus. In our experiments, we set the translation probability and lexical weight to 0.37. For a Chinese number or time phrase, which translation will be used, the translation generated by rules or extracted from parallel corpus, is decide by the translation model according to equation (3).

We use the Chinese-English named entity lists, which are released by LDC (Linguistic Data Consortium), as a dictionary to translate other named entities, such as the person names, organizations, places. Similar to the way deal with number and time phrases, we also put them into bilingual phrases table with the certain probability.

3.1.3. Decoder

Our decoder employs a beam search algorithm, similar to Pharaoh Decoder (Koehn, 2004). Since the distortion feature is not used in the translation model, we apply a monotone search and translate the input sentence from left to right.

Given a Chinese sentence, we first enumerate all possible words sequences, and check for each sequence if it has English translations in bilingual phrases table. Only the phrase with English translations is needed for decoding and selected into memory. For reducing the search space, in our experiments we select the top 20 English translations for each Chinese phrase according to the phrase translation probability $p(\widetilde{e}|\widetilde{f})$. During search, a phrase translation table is built as a cache, therefore, the huge bilingual phrases table can be stored on hard disk, and never needed to read into memory as a whole.

Then we search the best English translation in form of hypotheses, and store the hypotheses with stacks. At the beginning, an initial empty hypothesis is created, and no Chinese phrase is translated. Since we apply a monotone search, a Chinese words sequence spanning from 1 to j is selected, together with its possible English translations and the translation probabilities. Then a new hypothesis is generated from the initial empty hypothesis by computing the cost according to equation (3). The translated Chinese words are marked. Then another words sequence start from $j+1$ is selected. Finally, the cheapest hypothesis with all Chinese words covered is the output of the search.

Usually, the search space is too large to fit into memory, so pruning is very important for decoder. Two pruning methods are used during searching. The first one is recombination. Since all the hypotheses covered the same number Chinese words are stored in the same stack, if any two hypotheses in the same stack have the same language model state, e.g. the last two English words generated are equal if a trigram language model used, then the one with a lower probability (higher cost) will be discarded and cannot be extended in the following step. But the information should be kept for generating n-best translations. The second pruning method is histogram pruning. In a same stack, We call the hypothesis with higher probability the main-hypothesis and others with the same language model and lower probability the sub-hypothesis. Usually, a main-hypothesis may has a lot of sub-hypotheses, in our decoder, only the best n sub-hypotheses are stored. In addition, the size of stacks is fixed to a certain number, that is, only the top m main-hypotheses are kept. In the experiments, we set $n = 100$, $m = 100$.

In order to speed up decoding, future cost for each hypothesis is computed. We can estimate the future cost for any sequence of consecutive Chinese words by dynamic programming before decoding as done by (Koehn, 2004).

3.1.4. Discriminative training

There are many methods to train the parameters of log-linear model, such as GIS (Generalized Iterative Scaling) algorithm (Darroch and Ratcliff, 1972), Minimum Error Rate training (Och, 2003), etc. We use minimum error rate training to tune the weight of feature functions. We reimplemented Venugopal's trainer (Venugopal and Vogel, 2005) in C++, which runs faster than the original MATLAB version.

The feature weights λ are optimized in the following steps:

1. Initialize $\lambda_m = 1(m = 1, 2, \cdots, M)$
2. Compute an n-best list by performing search
3. Use the n-best list to do minimum error rate training, and get the new model parameters
4. Use the new model parameters to search, and get a new n-best list, which is combined with the previous n-best list.
5. goto step 3

The algorithm stops until the size of n-best list does not change, or the iteration reaches to a certain number. In our experiment, we set the size of n-best to 1000, which means 1000-best candidate translations are generated after each searching iteration. And the maximum iteration number is 10. In minimum error rate training algorithm, BLEU score (Papineni et al., 2002) is used to evaluate the translation quality, which is a very commonly used evaluation metric. The shortest reference sentence is used for the brevity penalty.

3.2. Lynx

Lynx is a decoder based on tree-to-string alignment template (TAT), which describes the alignment between a source parse tree and a target string. A TAT is capable of generating both terminals and non-terminals and performing reordering at both low and high levels. The TAT-based model is linguistically syntax-based because TATs are extracted automatically from word-alignment, source side parsed parallel texts. To translation a source sentence, we first employ a parser to produce a source parse tree and then apply TATs to transform the tree into a target string. More details can be found in (Liu et al., 2006).

We used seven feature functions analogous to default feature set of Pharaoh (Koehn, 2004). We extracted 1,792,025 TATs on 164K Chinese-English sentence pairs. The Chinese sentences were parsed with a Chinese parser developed by Xiong et al. (2005). The parser was trained on articles 1-270 of Penn Chinese Treebank version 1.0 and achieved 79.4% (F1 measure) as well as a 4.4% relative decrease in error rate.

For the task, we made use of bilingual phrases as special TATs. These bilingual phrases were the same with those used for our primary system. We did not tune the scaling factors on the official development set because it was difficult to parse the Chinese segments which contained several sentences without punctuation.

Although the second submission from Lynx used a 4-gram language model, it achieved lower BLEU score than the first submission because we did not change the scaling factors and the beam was smaller.

3.3. Bruin

Bruin is developed on the base of BTG (Wu, 1996) scheme. Throughout the whole translation process, we use merging rules to combine two consecutive blocks into a single larger block in the straight/inverted order, and use the lexical rule to translate source phrases into target phrases, which are restricted not to be null. The probability of lexical rule is calculated by phrase translation probabilities in both directions, IBM model 1 probabilities in both directions, word bonus and phrase bonus, and the language model probability in the log-linear form. The probability of merging rules is computed by the increment of the language model score and a special reordering model score in the log-linear form. The system Bruin1 used a maximum entropy reordering model which is described in (Xiong et al., 2006) in detail. The system Bruin2 used a flat reordering model that is related to the one by (Zens et al., 2004).

We developed a CKY-style decoder for Bruin system that employs a beam search algorithm, similar to the one by Chiang (2005). The decoder finds the best derivation that generates the input sentence and its translation. From the best derivation, the best English is produced. We use three ways to prune the search space. The first one is recombination. When two derivations in the same cell have the same w leftmost/rightmost words on the English yields, where w depends on the order of the language model, they will be recombined by discarding the derivation with lower score. The second one is the threshold pruning which discards derivations that have a score worse than b times the best score in the same cell. The last one is the histogram pruning which only keeps the top n best derivations for each cell. In all Bruin systems, we set n = 40, b = 0.5.

Catalog Number	Description
LDC2003E14	FBIS Multilanguage Texts
LDC2004T08	Hong Kong Parallel Text
LDC2002E18	Xinhua Chinese-English Parallel News Text Version 1.0 beta2
LDC2004T07	Multiple Translation Chinese Part3
LDC2005T06	Chinese News Translation Text Part1
LDC2003E07	Chinese Treebank English Parallel Corpus

Table 2: Training Data List

4. Experiments

In 15 days evaluation, we do experiments on two tasks: the ASR and Verbatim task of Chinese Mandarin to English translation. Confucius and Bruin tune the model parameters on the development set released by TC-STAR, and Lynx set the parameters depend on experiences. All systems drop the unknown words (Koehn et al., 2005) of translations.

4.1. Training Data

We use about 2.4M sentence pairs with about 60M Chinese words and 67M English words to extract bilingual phrases, which are come from the corpus released by LDC. See Table 2 for details. Please notice that, for LDC2002E18 we only use the first 15k sentence pairs, for LDC2004T08 we only use the HK_Hansards and HK_News. In addition, the corpus LDC2003E01 (Chinese-English Name Entity Lists version 1.0 beta) is used as a Named Entity dictionary in Confucius.

We use SRI Language Modeling Toolkit (Stolcke, 2002) to train language model with modified Kneser-Ney smoothing (Chen and Goodman, 1998). We train three language models, a trigram model and a 4-gram model on the training corpus, and a 4-gram model on Xinhua portion of Gigaword with about 190M words. Since our computer (4GB multiprocessor Linux machine) cannot handle such a large data, we divide the Xinhua corpus into 4 parts, and train 4-gram language model separately. Then the 4 parts are interpolated with the weight 0.25.

System	Language Model	Named Entity	DEV06_ZHEN_VERBATIM	TEST06_ZHEN_VERBATIM
Confucius1	4-gram Xinhua model	Used	0.1574	0.1378
Confucius2	3-gram Training data model	Used	0.1496	0.1324
Lynx1	3-gram Training data model	No	/	0.1288
Lynx2	4-gram Xinhua model	No	/	0.1107
Bruin1	3-gram Training data model	No	0.1537	0.1373
Bruin2	3-gram Training data model	No	0.1521	0.1255

Table 3: BLEU-4 Scores on development set and test set of Verbatim task. Named Entity used means LDC2003E01 (Chinese-English Name Entity Lists version 1.0 beta) used as a Named Entity dictionary. Note that Lynx didn't use the official development set to tune parameters.

System	Language Model	Named Entity	DEV06_ZHEN_ASR	uka-limsi_eval06_bn_zh_ public_primary.sys1.segmented
Confucius1	4-gram Xinhua model	Used	0.1523	0.0971
Confucius2	3-gram Training data model	Used	0.1422	0.0936
Bruin2	3-gram Training data model	No	0.1403	0.0883

Table 4: BLEU-4 Scores on development set and test set of ASR task. Named Entity used means LDC2003E01 (Chinese-English Name Entity Lists version 1.0 beta) used as a Named Entity dictionary. Note that the Bruin2 system did not finish the minimum error rate training due to the time pressure. So the result is not finalized.

4.2. Results

About 97M bilingual phrases are extracted from training data, 5M are used for ASR task and 5M for Verbatim task. We use the tools mteval-v11b.pl supplied by TC-STAR to evaluate the translation quality on development data. All of the three systems (Confucius, Lynx and Bruin) participate in the Verbatim task, results are shown in Table 3. We can see that the performance of Bruin1 is better than Confucius2 used the same language model; part of the reason is that Bruin1 use a reordering model based on maximum entropy, and this model also achieves a higher score than flat reordering model used in Bruin2 especially on test data.
Confucius1 used a 4-gram language model trained on Xinhua portion of Gigaword and get a significant improvement, which indicate that a good language model can help us to improve translation quality. Because of the time pressure, Lynx only use 164K sentence pairs of the training data to extract tree-to-string alignment template, additionally, as described in section 3.2, it didn't tune model parameters, so the translation quality seems not very well.
System Confucius and Bruin participate in the ASR task, results are shown in Table 4. We choose the results generated by Confucius1 as our primary submission for both the tasks.

5. Conclusion

In this paper we describe our systems participated in the 2006 TC-STAR Run #2 SLT Evaluation. We developed three systems based on different techniques, and achieved good results.

Acknowledgements

We would like to thank the teachers of our laboratory: Shouxun Lin, Yueliang Qian, Yajuan Lv, as well as other members of our team: Shiqi Cui, Weihua Luo, Lei Fu, Jin Huang, Like Huang, and Haotao Mi. This work was supported in part by National High Technology Research and Development Program under grant #2005AA114140 and National Natural Science Foundation of China under grant #60573188.

6. References

Stanley F. Chen and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Harvard University Center for Research in Computing Technology.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of ACL 2005*, pages 263–270.

J.N. Darroch and D. Ratcliff. 1972. Generalized iterative scaling for log-linear models. *Annals of Mathematical Statistics*, 43:1470–1480.

Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of HLT-NAACL 2003*, pages 127–133.

Philipp Koehn, Amittai Axelrod, Alexandra Birch Mayne, Chris Callison-Burch, Miles Osborne, and David Talbot. 2005. Edinburgh system description for the 2005 iwslt speech translation evaluation. In *International Workshop on Spoken Language Translation*.

Philipp Koehn. 2004. Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In *Proceedings of the Sixth Conference of the Association*

for Machine Translation in the Americas, pages 115–124.

Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of the 44th Annual Meeting of the ACL*.

Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting of the ACL*, pages 440–447.

Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual Meeting of the ACL*, pages 295–302.

Franz Josef Och. 2002. *Statistical Machine Translation: From Single-Word Models to Alignment Templates*. Ph.D. thesis, Computer Science Department, RWTH Aachen, Germany.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the ACL*, pages 160–167.

K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association of Computational Linguistics (ACL)*.

Andreas Stolcke. 2002. Srilm – an extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken language Processing*, volume 2, pages 901–904.

Ashish Venugopal and Stephan Vogel. 2005. Considerations in maximum mutual information and minimum classification error training for statistical machine translation. In *Proceedings of the Tenth Conference of the European Association for Machine Translation (EAMT-05)*.

Dekai Wu. 1996. A polynomial-time algorithm for statistical machine translation. In *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*.

Deyi Xiong, Shuanglong Li, Qun Liu, Shouxun Lin, and Yueliang Qian. 2005. Parsing the penn chinese treebank with semantic knowledge. In *Proceedings of IJCNLP 2005*, pages 70–81.

Deyi Xiong, Qun Liu, and Shouxun Lin. 2006. Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of the 44th Annual Meeting of the ACL*.

R. Zens, H. Ney, T. Watanabe, and E. Sumita. 2004. Reordering constraints for phrase-based statistical machine translation. In *Proceedings of the 20th International Conference on Computational Linguistics (CoLing 2004)*, pages 205–211, Geneva, Switzerland.